پروژه نهایی هوش محاسباتی

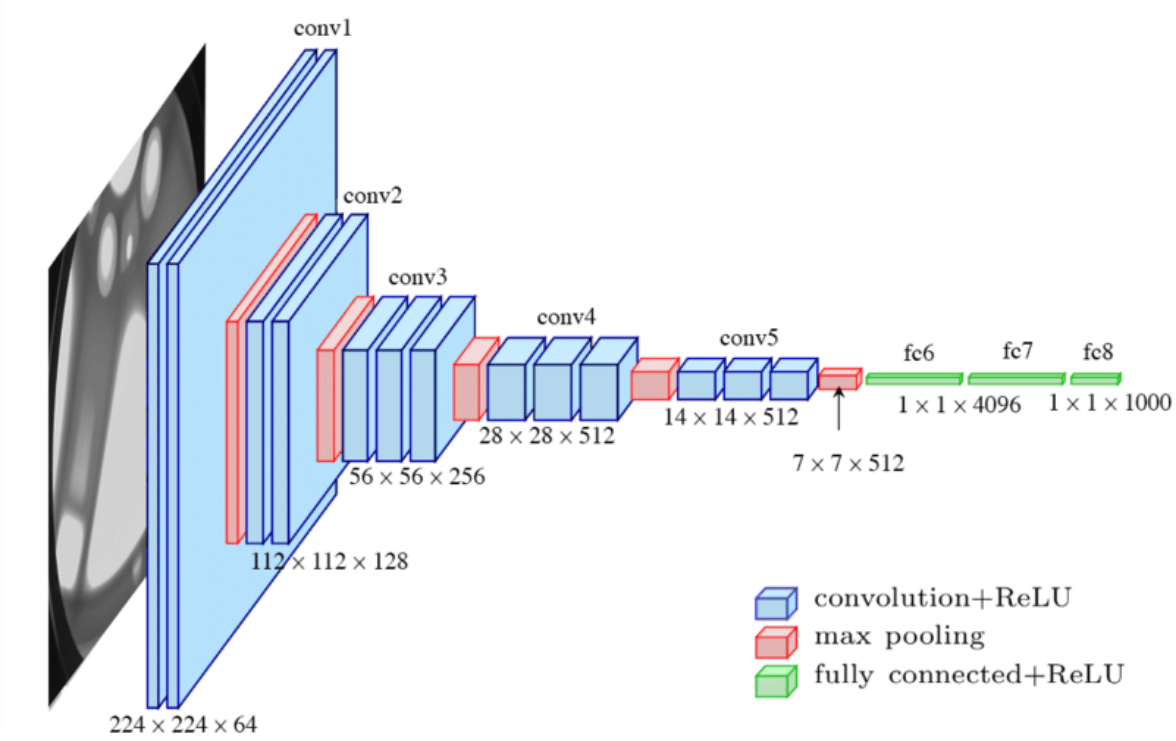
1.شقایق بوستان بخش(**9612268201**)

2.سارا اسدی(**960122681016**)

در این پروژه قصد داریم با استفاده از شبکه عصبی convolution تصاویر سگ و گربه را طبقه بندی کنیم و مدلی را در راستای دسته بندی سگ و گربه ارائه دهیم.

برای طراحی این مدل از سیستم انتقال یادگیری یا همان Tranfer learning

با استفاده ازpre-trained vgg16 Network پیاده سازی شده است,در این پروژه کتاب خانه های Keras , Tensorflo و Sklearn که از سریع ترین و قوی ترین پلتفرم ها در زمینه یادگیری ماشین هستند به کار گرفته شده اند.

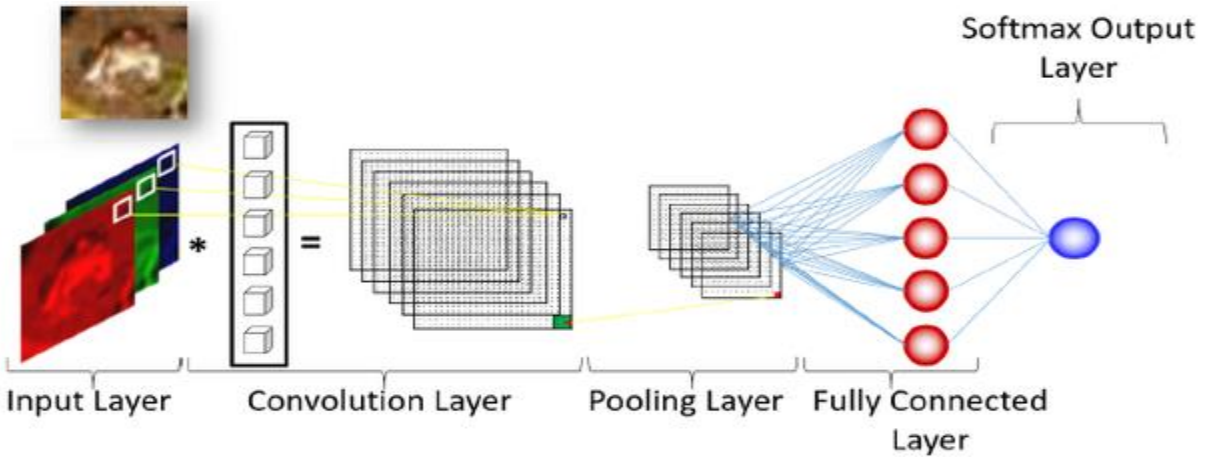


در قسمت اول شبکه ، پنج بلوک کانوشنال (**conv1** تا **5**) را مشاهده می کنیم ، که در لایه های محکم جمع شده قرار دارند و به دنبال آن یک لایه جمع کننده حداکثر بنابراین ما این قسمت را**part convolutional**می نامیم.

در این پروژه علاوه بر استفاده از داده های از پیش Train شده vgg16 دیتاست دستی تهیه کردیم که شامل 100 عکس گربه و 100 عکس سگ میباشد.

ما برای هر کدام از عکس ها 3 ماتریس 224 x224 به رنگ های red,blue,green(RGB) داریم که همه تصاویر را تغییر شکل داده و آنها را به صورت یک آرایه NumPy ذخیره کردیم.

این دیتاست در مرحله Train_test_split از هم جدا میشوند.




```
[ ]  X = cats_img + dogs_img
     y = [0] * len(cats_img) + [1] * len(dogs_img)
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=True)
     X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, test_size=0.5, shuffle=True)
     print(len(X_train), len(y_train), len(X_val), len(y_val), len(X_test), len(y_test))
```

```
160 160 20 20 20 20
```

```
X_train = np.array(X_train)
y_train = np.array(y_train)
X_val = np.array(X_val)
y_val = np.array(y_val)
X_test = np.array(X_test)
y_test = np.array(y_test)
print('Train :', X_train.shape, y_train.shape)
print('Val :', X_val.shape, y_val.shape)
print('Test :', X_test.shape, y_test.shape)
```

```
Train : (160, 224, 224, 3) (160,)
Val : (20, 224, 224, 3) (20,)
Test : (20, 224, 224, 3) (20,)
```

در این بخش ، ما می خواهیم یک الگوی شبکه عصبی کانونشنال پایه را برای مجموعه داده سگها در مقابل گربه ها تهیه کنیم.

یک نقطه شروع خوب ، اصول معماری کلی مدل های VGG است , زیرا آنها در رقابت ILSVRC 2014 به بهترین عملکرد دست یافته اند و به دلیل ساختار ماژولار معماری درک و اجرای آن آسان است.

این کار را می توان با بارگیری مدل VGG-16 ، برداشتن لایه ی متصل از انتهای خروجی مدل و Flatten کردن محتویات آن و سپس اضافه کردن لایه های متصل به جدید به خروجی مدل انجام داد.

لایه های جدید شامل یک لایه Dense با 1024 نورون و یک لایه Dense با یک نورون , تابع های فعال ساز ما relu و sigmoid هستند ,تابع فعال ساز sigmoid برای گرفتن خروجی به صورت احتمال در نظر گرفته شده. در این حالت آموزش زیادی لازم نخواهد بود ، زیرا فقط آخرین لایه جدید دارای وزن قابل آموزش است و بقیه لایه ها Freeze شده اند. به این ترتیب تعداد دوره های آموزشی را در 20 حل خواهیم کرد(بهینه ساز این مدل Adam میباشد).

```
[ ]  model = VGG16(include_top=False, weights='imagenet', input_shape=(224, 224, 3))
     for each_layer in model.layers:
         each_layer.trainable = False
     interaction_output = model.layers[-1].output
     x = keras.layers.Flatten()(interaction_output)
     x = keras.layers.Dense(1024, activation="relu")(x)
     x = keras.layers.Dense(1, activation="sigmoid")(x)
     model = keras.Model(model.input, x)
     model.summary()
```

```
[ ] checkpoint = keras.callbacks.ModelCheckpoint("/content/drive/My Drive/BCI982/model.h5", monitor='val_acc', save
    model.compile(optimizer="adam", loss="binary_crossentropy", metrics=['accuracy'])
    model.fit(X_train, y_train, epochs=20, validation_data=(X_val, y_val), batch_size = 16, callbacks=[checkpoint]
    model.save('/content/drive/My Drive/BCI982/mainModel.h5' )
```

در مرحله بعد ، مدل را Fit می کنیم .

میتوانیم مدل را بیازماییم , Result نهایی بدست آمده از Train 97% و Test 100% میباشد.

```
_, acc = model.evaluate(X_train, y_train)
print('Train Accuracy: %.2f' % (acc*100))
_, acc = model.evaluate(X_test, y_test)
print('Test Accuracy: %.2f' % (acc*100))
```

```
160/160 [==============================] - 10s 62ms/step
Train Accuracy: 97.50
20/20 [==============================] - 2s 114ms/step
Test Accuracy: 100.00
```

سپس تابع پیش بینی () را فراخوانی می کنیم تا محتوای موجود در تصویر را به صورت عددی بین "0" و "1" برای "گربه" و "سگ" پیش بینی کند.